

El Robot Scribbler

Manual de ayuda



***INGENIERIA DE MICROSISTEMAS
PROGRAMADOS S.L.***

MSE

**Microsystems
Engineering**

C/ Alda. Mazarredo Nº 47 - 1º Dpto. 2
48009 BILBAO - BIZKAIA
Tel/Fax: 94 4230651

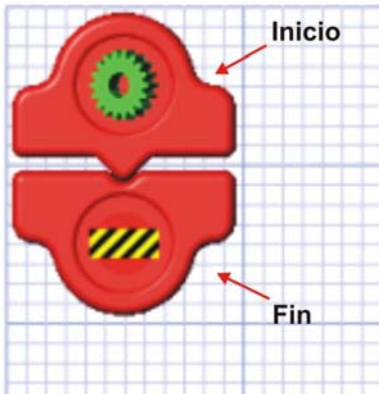
email: info@microcontroladores.com
www.microcontroladores.com

El robot Scribbler

INDICE GENERAL	1
1.- COMENZANDO	3
1.1 El nuevo espacio de trabajo	3
1.2 Navegando por el espacio de trabajo	3
1.3 Insertando una pieza	3
1.4 Cargando el programa del robot	4
1.5 Editando un programa	5
1.6 Salvando un programa	7
1.7 Creando el espacio de trabajo	7
1.8 Cargando un programa	7
1.9 Cargando el programa de fábrica	7
1.10 Calibrando el Scribbler	8
1.11 Monitorizando los sensores	8
1.12 Ayuda	8
2.- LAS PIEZAS DE ACCION DEL PROGRAMA	9
2.1 La pieza de control de LEDS (Bombillas)	9
2.2 La pieza de control de movimiento	9
2.3 Pieza de acción de pausa	10
2.4 Pieza de acción de sonido	10
2.5 Pieza de avisos o banderas	11
2.6 Pieza de llamada a subrutina	12
2.7 Pieza de observación de sensores	12
2.8 Pieza de cálculo	13
3.- PIEZAS CONDICIONALES	13
3.1 Si-entonces	13
3.2 En caso contrario	14
3.3 Y-si	15
3.4 Condición de aviso o por bandera	15
3.5 Condición del sensor de líneas	16
3.6 Sensor de códigos de barras	16
3.7 Detector de obstáculos	17
3.8 Detector de choque	17
3.9 Condición del sensor de luz	17
3.10 Lanzamiento de moneda	18
3.11 Pulsación del botón de RESET	19
4.- BUCLES	19
4.1 Repetición tipo Loop	19
4.2 Etiquetas de salida	20
5.- SUBRUTINAS	20
5.1 Creando subrutinas	20
6.- SENCILLO EJEMPLO DE RASTREO	22

1.- Comenzando

1-1 El nuevo espacio de trabajo



dejamos para más adelante.

Figura 1

Cuando se ejecuta el editor gráfico de programas del Scribbler, comenzamos a trabajar en un nuevo espacio de trabajo. El espacio de trabajo es el lugar donde incluiremos nuestras piezas o bloques del programa Scribbler. El programa principal siempre empieza con una pieza roja de inicio con una rueda dentada verde en su interior (véase la figura 1). Nuestros programas siempre deberán acabar con la pieza de fin, una pieza roja con una señal de barrera en su interior. El programa de esta figura, es un programa vacío que no hace nada, pues según empieza, acaba. Si cargáramos este programa en el scribbler, el scribbler no haría nada, permanecería inmóvil.

Para hacer que el Scribbler haga algo, tenemos que añadir piezas a este programa. Las piezas se seleccionan usando los botones de la barra vertical que tenemos en la izquierda del programa Scribbler. Estas piezas siempre se tienen que insertar entre otras dos que existan previamente. La única excepción la haremos cuando creamos una subrutina, pero eso lo

1-2 Navegando por el espacio de trabajo

A menudo, mientras creamos nuestro programa, éste resulta más grande que la ventana de trabajo. Afortunadamente, existen muchas formas de navegar por el mismo. La más obvia, por supuesto, son las barras de scroll, en la parte inferior y derecha de la ventana. También disponemos de un sistema de scroll automático. Cuando acercamos el ratón a cualquiera de los bordes de nuestra área de trabajo, éste cambiará a una pequeña flecha con un rectángulo en su extremo (ver la figura 2). Al de un breve tiempo, el área de trabajo empezará a desplazarse en sentido contrario al de la flecha. Esto ocurre hasta que llegamos el final del área de trabajo o movemos el ratón separándolo del borde.

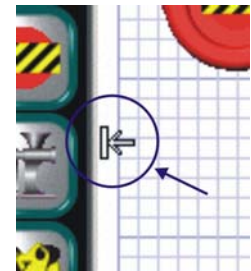


Figura 2



Figura 3

También existe un botón de zoom (uno con una lupa de aumento, o a veces también, una lupa para alejar la imagen). Haciendo clic en este botón, todo nuestro área de trabajo va a cambiar de tamaño para ajustarse al tamaño de la ventana. En cuanto pinchemos en algún elemento del programa, la ventana volverá a su tamaño normal.

No debemos preocuparnos nunca por llegar al final del área de trabajo, ya que será muy fácil reacomodarlos todo o simplemente trabajar más abajo.

1-3 Insertando una pieza

Para insertar una pieza o bloque en nuestro programa, primero deberemos seleccionar dicha pieza en la barra de herramientas de la izquierda de la pantalla. Hacemos clic en esos botones y se iluminará el botón seleccionado. En la figura 4 se ha seleccionado la herramienta de encendido y apagado de los leds (luces del robot).

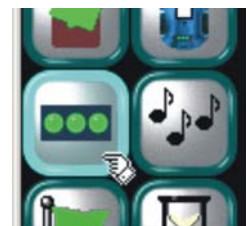


Figura 4

Cursor de inserción

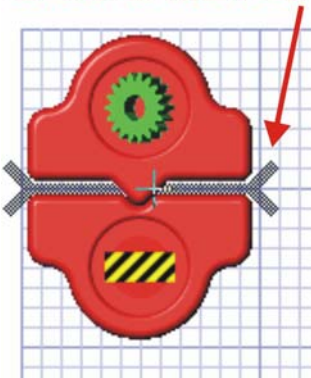


Figura 5

Una vez posicionado dónde queremos introducir la pieza y, una vez que ha aparecido el cursor, hacer clic con el ratón. En ese momento, la pieza que hemos seleccionado quedará fijada formando parte de nuestro programa actual. En la mayoría de las ocasiones, según la pieza seleccionada, el programa mostrará una ventana de configuración que nos permitirá configurar la acción asociada a la pieza recién introducida. En este caso, haciendo clic en los interruptores que aparecen en pantalla, podremos configurar qué leds, o bombillitas, queremos que se enciendan o se apaguen. En la figura 6, hemos apagado 2 bombillas y encendido una (la de la izquierda).

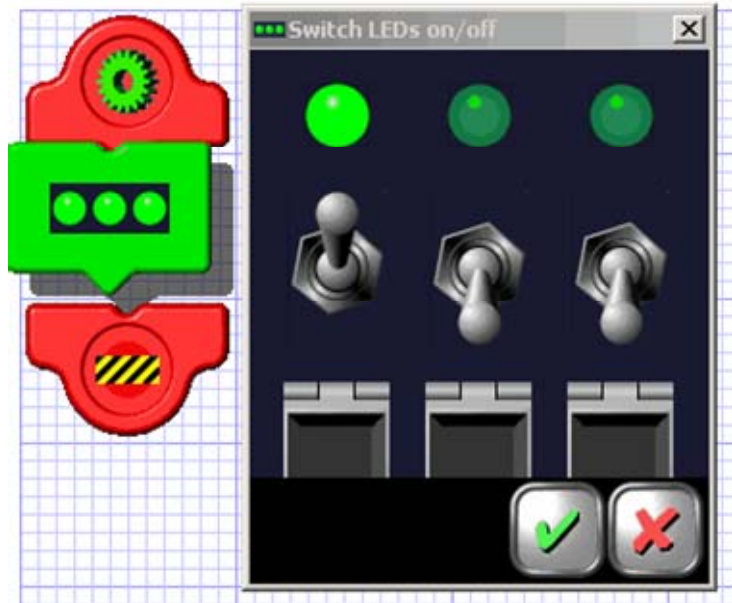


Figura 6

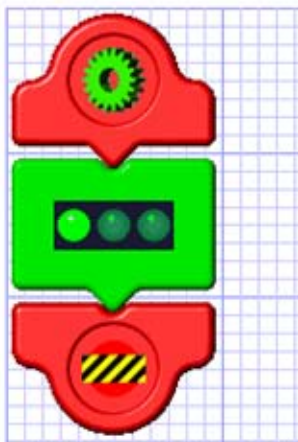


Figura 7

A continuación, pulse en el botón con el tic verde de aceptar. Cuando lo haga, la pieza se insertará en su sitio, debidamente configurada y reflejando la acción que se va a realizar. Si ha seguido estos pasos, usted ya habrá confeccionado su primer programa, el de la figura 7, cuya misión es encender un LED.

1-4 Cargando el programa en el robot

Para probar nuestro Nuevo programa, tendremos que conectar el Scribbler al puerto serie de nuestro ordenador. Una vez hecho esto, encienda el interruptor de encendido del robot. Sujete el robot cuando realice esta acción porque de lo contrario puede que el robot se ponga a andar. Esto se puede deber a que el robot contiene algún otro programa previo. A continuación haga clic en el botón de carga de programa, el mostrado en la figura 8. Su programa será transferido al Scribbler. Cuando finalice dicha transferencia el programa empieza a ejecutarse de forma automática. En este caso usted debería ver como el LED izquierdo se ilumina.



Figura 8

1-5 Editando un programa



Figura 9

Supongamos que realmente, lo que queríamos era encender el LED de la derecha. ¡No hay problema! Podemos añadir, cambiar o borrar piezas en nuestro programa a nuestro gusto.

Existen varias formas de seleccionar una o varias piezas:

1. Podemos activar el botón de selección (el del dedo índice), y hacer clic en la pieza que queramos seleccionar. Podemos seguir señalando piezas para añadirlas a la selección. Si ha señalado demasiadas, puede marcar o desmarcar haciendo uso del doble clic. Tan solo existe una limitación, y es que no podemos marcar una única pieza perteneciente a un bloque. Por ejemplo, si marcamos el inicio del programa, como pertenece al bloque programa, se nos marcaría todo el programa. Esto se debe a que el scribbler no permite, para evitar errores, que haya bloques incompletos.
2. Podemos pinchar con el botón derecho en cualquier pieza para seleccionarla. Aparece una nueva ventana con las opciones del menú de edición asociadas a esa pieza.

Una vez que tenemos seleccionada una pieza o un grupo de piezas, podemos pinchar con el botón derecho sobre ellas para acceder al menú de edición. El menú contiene botones para copiar, cortar y borrar. Cada una de estas opciones se iluminarán al ser seleccionadas. Para cancelar una selección, simplemente haga clic de nuevo en ella. Para realizar la operación deseada pulse en el tic verde de aceptar.

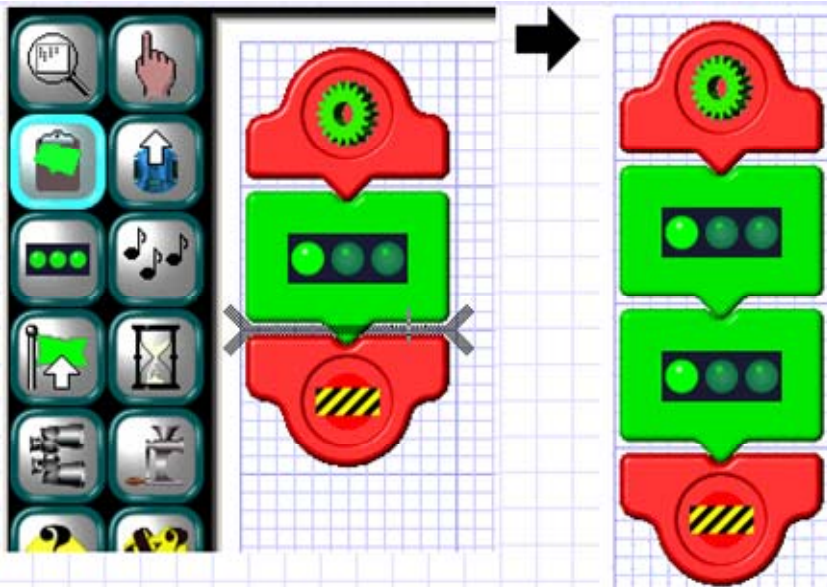
Si hemos seleccionado una sola pieza o un único grupo sintáctico de piezas, tales como un ciclo loop, o una condicional (estudiadas más adelante), también existe un botón en la parte izquierda de la pantalla que muestra el tipo de pieza o grupo seleccionado. Cuando hacemos clic en este botón, la ventana original de edición de esta pieza o grupo aparece admitiendo el cambio de su contenido.

Para este ejemplo, en vez de editar la pieza que habíamos introducido, vamos a copiar y pegarla. En primer lugar la seleccionamos con el botón derecho y pinchamos en el botón Copy del menú de edición. Después pulsaremos el tic verde de aceptar (figura 10).



Figura 10

↑
ACEPTAR



Con el objeto copiado, movemos el cursor del ratón justo debajo de la pieza original de manejo de LEDs y hacemos clic. Una copia de nuestra pieza se insertará justo en la posición indicada.

Figura 11

Pulemos con el botón derecho en el objeto recién copiado para abrir el menú de edición. Pero esta vez pulsamos el botón de LEDs (como en la figura 12) y aparecerá en pantalla las posibles configuraciones de los leds.

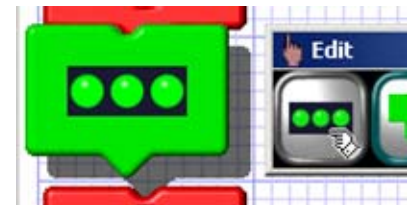
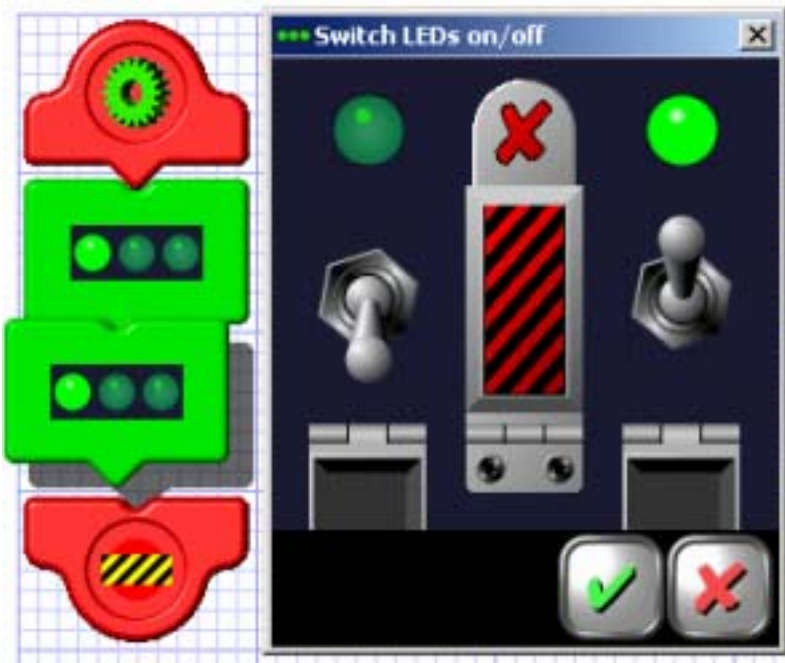


Figura 12



Ahora puede cambiar los interruptores para cambiar el LED que queremos encender o apagar. En el ejemplo de la figura 13, el LED izquierdo está apagado, el de en medio permanece sin cambios y el derecho está encendido.

Figura 13

Una vez que pinchemos en el tic verde de aceptar el programa resultante será el mostrado en la figura 14.



Figura 14

Si cargásemos este programa en el Scribbler, seguramente solo veríamos el LED derecho encendido. El led izquierdo se encendería durante tan poco tiempo que sería imperceptible a la vista.

Por tanto, seguramente deseará eliminar la pieza que enciende el led izquierdo. Para hacerlo pinche con el botón derecho sobre dicha pieza para obtener el menú de edición y pulse sobre el botón de eliminar (la papelera mostrada en la figura 15). También podría hacerlo con la opción cortar (tijeras), pero esto sería más adecuado si luego lo quisiéramos pegarlo en otra ubicación.



Figura 15

1.6 Salvando el programa

En esta figura 16 podemos observar el resultado tras eliminar la pieza anterior. Podemos salvar el programa en el disco duro para poder recuperarlo más tarde. Haciendo clic el botón señalado podremos poner nombre a nuestro programa y guardarlo donde queramos.

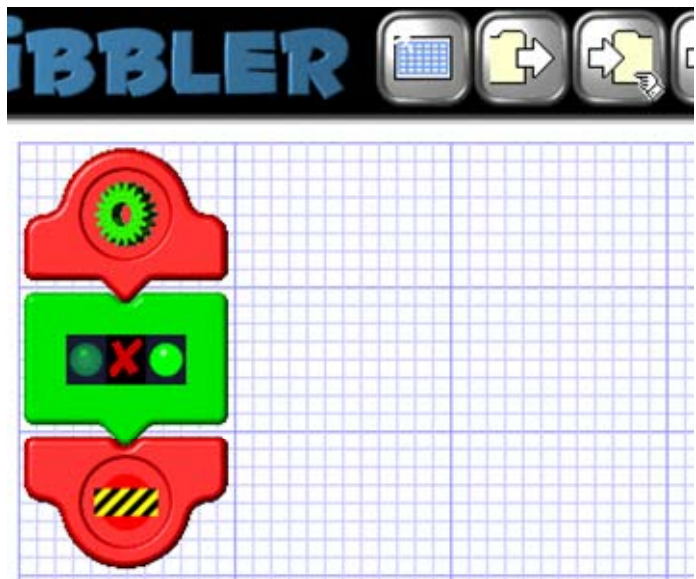


Figura 16

1.7 Creando el espacio de trabajo



Una vez guardado el programa, podemos limpiar la zona de trabajo y empezar uno nuevo. Simplemente pinchamos en el botón de nuevo área de trabajo, figura 17.

Figura 17

1.8 Cargando un programa

Para cargar la versión guardada de un programa, simplemente pinchamos en el botón de carga de programa (el mostrado en la figura 18) y se abrirá una pantalla donde podremos elegir el programa deseado.



Figura 18

1.9 Cargando el programa de fábrica



Para restaurar el programa de fábrica con el que viene el scribbler, conectaremos el cable serie en el puerto de programación del robot y encenderemos el scribbler. Pulsaremos en el botón de restaurar (figura 19). A continuación el programa de fábrica del robot se cargará en el mismo. Esta operación no tiene ningún efecto sobre el programa que estemos creando en nuestra área de trabajo.

1.10 Calibrando el Scribbler



El Scribbler viene de fábrica con los motores calibrados y programados. Los sensores de luz, sin embargo, deben ser calibrados por el usuario. Es un proceso muy simple. Primero, cargue el programa de calibración usando el botón de calibración (figura 20). A continuación, coloque el robot sobre una superficie dura y plana. Entonces presione el botón de reset 3 veces, en una rápida sucesión, y entonces el robot emitirá una breve melodía y empezará a moverse unas 5 o 6 veces.

(Si la habitación es muy oscura, se parará antes y emitirá un pitido. Esto es debido a que el scribbler rota según la intensidad de luz que encuentra. Esa rotación se detiene cuando los sensores de luz se han calibrado y entonces suena un "Ta Da" para indicar que se ha acabado el proceso). Esta operación no tiene ninguna repercusión sobre el programa que estemos editando.

1.11 Monitorizando los sensores

Si tiene instalado el programa opcional de monitorización de sensores en su ordenador, el botón monitor (figura 21) le dará acceso a dicho programa. Simplemente haga clic en dicho botón.



Figura 21

1.12 Ayuda

Para acceder a la ayuda, simplemente pinche en el botón de la figura 22.

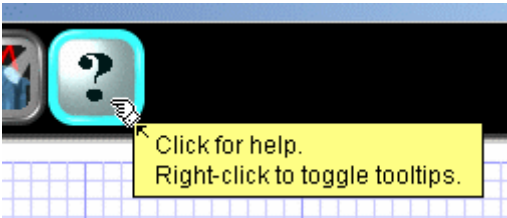


Figura 22

2.- Las piezas de acción del programa

Las piezas más utilizadas, son las piezas de acción del programa. Son piezas cuyo comportamiento se puede configurar y que hacen que el robot actúe de determinada manera. Todas las piezas de acción son de color verde. Y cada acción posible se encuentra en la barra de herramientas situada en la izquierda de la pantalla.

2.1 La pieza de control de LEDS (Bombillas)



Es la que hemos utilizado en los ejemplos anteriores (figura 23) y permite controlar el estado de los tres leds que dispone el robot. Como ya se visto, cada led individual se puede encender, apagar o que no cambie de estado.

Figura 23

2.2 La pieza de control de movimiento

La pieza de control de movimiento la usaremos para mover o parar el robot. Puede ser tan simple como lo representado en el primer ejemplo de la figura 24 (muévete hacia delante tres cuartos de vuelta) o más compleja, como el segundo ejemplo (rueda a la derecha a media velocidad durante 1.15 segundos, después para y llama a la subrutina amarilla).



Figura 24



Figura 25

La ventana de edición de la pieza de control de movimiento, mostrada en la figura 25, nos permite ajustar la velocidad del Scribbler de distintas maneras. También podemos indicar cuánto tiempo queremos que se mueva el robot. Ese tiempo se puede elegir en fragmentos de 0,05 segundos, hasta un máximo de 5 segundos. Tras el tiempo indicado el robot se parará y si no indicamos ningún tiempo (tiempo = 0) entonces el robot no parará nunca (hasta que lo apaguemos o se le acaben las pilas).

Hay tres tipos de movimiento, **Establecer** (mostrado como “=”), **Incrementar** (+), y **Multiplicar** (X). Estos se eligen desde la ventana de selección de tipo de movimiento. Incrementar, incrementa las velocidades de las ruedas izquierda y derecha hasta un máximo y mínimo de +100 o -100. Multiplicar, multiplica la velocidad de las ruedas en un rango desde -1,25 hasta 1,25. Si usamos incrementar o multiplicar no podemos usar tiempo porque se supone que estamos en una opción en la que el robot está continuamente en movimiento.

Como muchas otras piezas, esta nos permite llamar a subrutinas una vez que la acción programada se ha completado. La selección de la subrutina se hace mediante colores. En botón de selección de subrutinas (lleva un engranaje en su interior) va cambiando de colores secuencialmente. Cada color representa una subrutina distinta.

2.3 Pieza de Acción de pausa



Figura 26

La acción de pausa mostrada en la figura 26 se usa para suspender el programa del Scribbler temporalmente por periodos desde 0,001 segundos hasta 60 segundos. No cambia el estado del Scribbler, sin embargo, si el scribbler se está moviendo, continuará así durante dicho periodo. También los leds mantendrán el estado actual durante la pausa.

Cuando se edita la acción de pausa, aparece un reloj de arena como el mostrado en la figura 27. El tiempo a permanecer suspendido, lo elegiremos mediante la barra de scroll vertical.



Figura 27

2.4 Pieza de Acción de sonido



Figura 28

La acción de sonido mostrada en la figura 28 se emplea para activar el altavoz interno del Scribbler con el que se pueden generar pequeñas melodías, decir palabras en código Morse, etc. El programa viene con una librería de sonidos variados. Así mismo, al finalizar la acción del sonido podemos llamar a una subrutina.

Cuando tratamos de editar la acción del sonido, obtenemos una pantalla como la mostrada en la figura 29. Los sonidos predeterminados se copian a la acción de sonido simplemente haciendo clic sobre ellos. Si pulsamos con el botón derecho y lo mantenemos pulsado sobre un sonido predeterminado, podremos escucharlo antes de programarlo en nuestro robot. Para borrar un sonido, primero pincharemos sobre él y luego pulsaremos en el botón de la papelera.

Una vez establecida la secuencia de sonidos, puede escuchar la secuencia entera pulsando el botón PLAY. El botón PLAY se iluminará mientras el sonido se esté reproduciendo. Aunque en el PC podemos controlar el volumen, esta acción no repercute para cuando se reproduzca en el robot, pues siempre lo hará al mismo volumen.

Importante: Use los sonidos con moderación. Demasiados sonidos pueden hacer que se agote la memoria del robot.

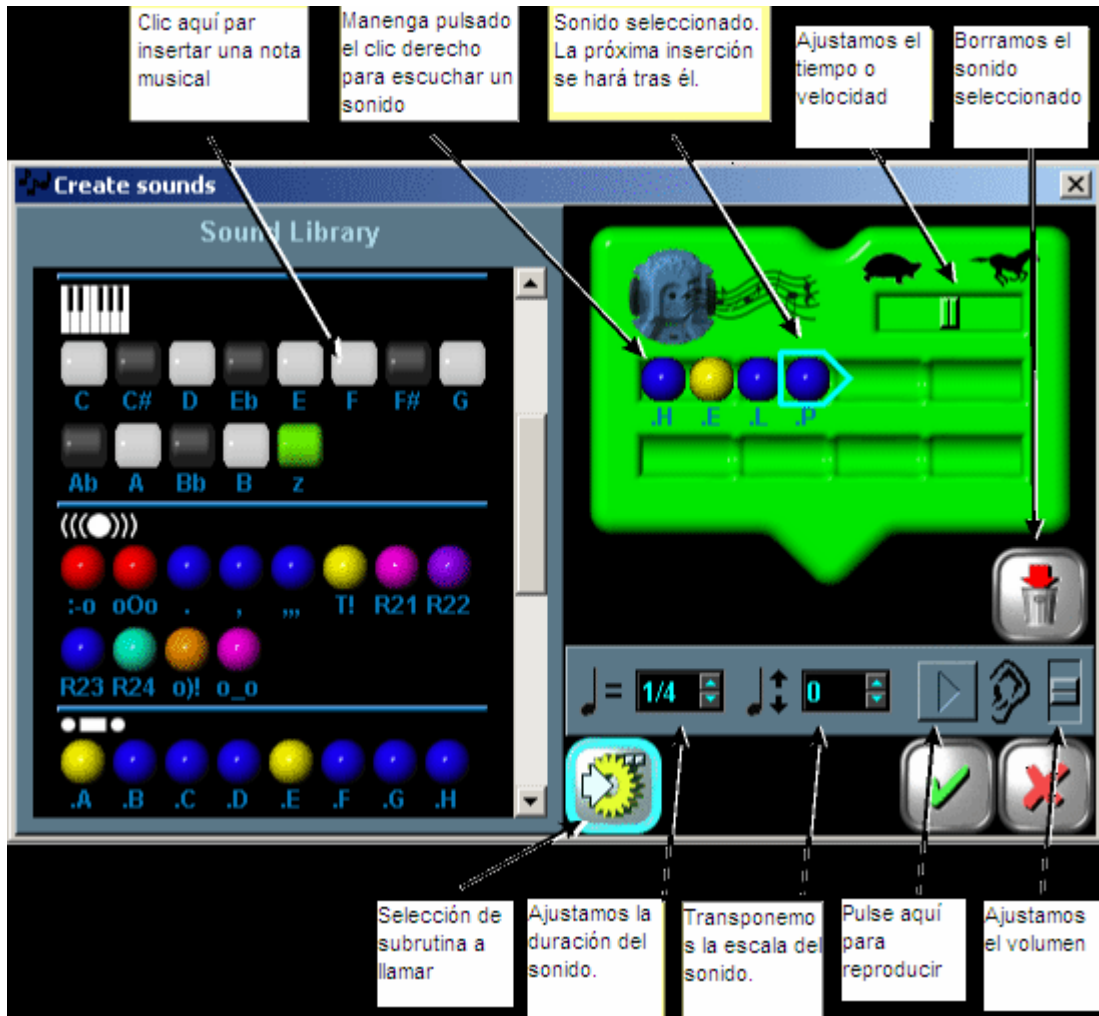
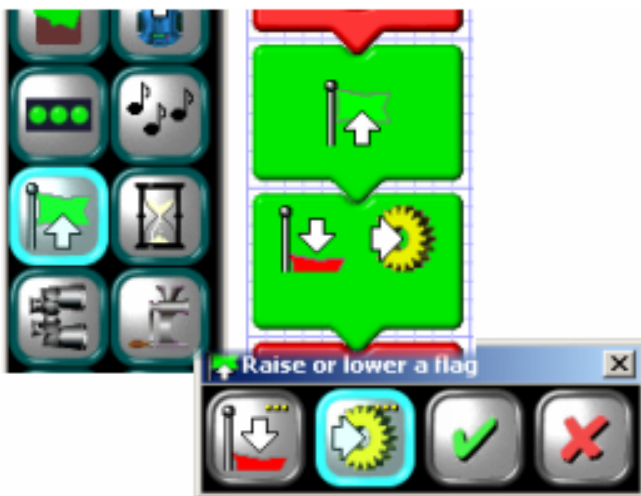


Figura 29

2.5 Pieza de avisos o banderas



Esta pieza (figura 30) se usa para pasar avisos o señales a otras partes del programa cuando sucede algún evento. Hay 7 banderas de distintos colores. Con ellas indicaremos de que algo se ha alcanzado (bandera subida) o no (bandera bajada). Más tarde veremos como usar piezas condicionales para analizar el estado de una bandera y en función de dicho estado actuar de una u otra manera. También pueden llamar a subrutinas.

La ventana de edición de la pieza de avisos es esta. Tiene botones de elección múltiple para la acción y para decidir a que subrutina llamar.

Figura 30

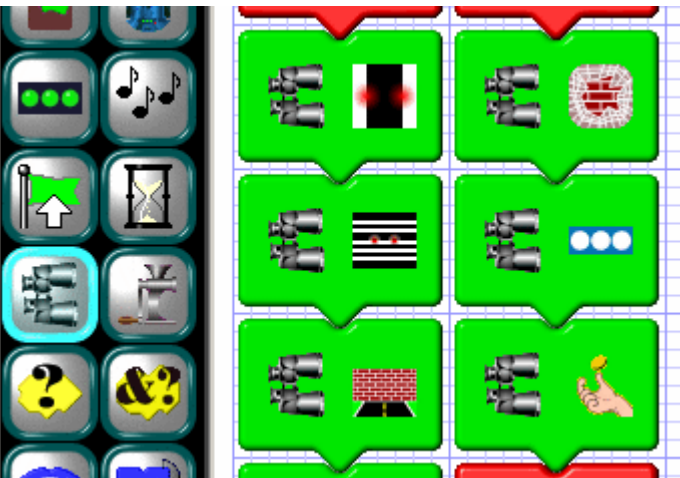
2.6 Pieza de llamada a subrutina



La pieza de llamada mostrada en la figura 31 se usa para llamar a una subrutina. Una subrutina es un conjunto de acciones que deben ejecutarse varias veces a lo largo de un programa, pero que solo se escriben una vez. Las subrutinas se identifican por el color del engranaje que hay en el interior de la pieza. Así una llamada a subrutina con engranaje de color rojo, ejecutará todas las instrucciones contenidas entre las piezas de inicio y fin de la subrutina del mismo color. Su ventana de edición de pieza tiene un botón de elección múltiple, representado por el engranaje, en el que se pueden seleccionar secuencialmente entre siete colores para identificar la subrutina a la que se llama.

representado por el engranaje, en el que se pueden seleccionar secuencialmente entre siete colores para identificar la subrutina a la que se llama.

2.7 Pieza para la observación de los sensores (análisis de los sensores)



Esta pieza se utiliza para preguntar al robot sobre el estado de uno de sus sensores para usarlo posteriormente en una condición. Hay 6 sensores que pueden observarse independientemente. Según la figura 32, empezando por la primera columna de la izquierda y de arriba abajo, estos son : sensor de línea, sensor de códigos de barra, sensor de obstáculos. En la segunda columna de la derecha, de arriba abajo tenemos: sensor de choque, sensor de luz y tiro de la moneda. Cada uno de estos sensores se estudiará más adelante en la sección dedicada a las piezas condicionales.

Una vez realizada una observación, su resultado persiste hasta que se hace otra. Así podemos usar esa misma observación en múltiples condiciones.

Figura 32

En algunos casos analizar un sensor con esta pieza de observación de sensores no es necesario. Esto es debido a que el propio software del Scribbler las inserta automáticamente cuando considera que son necesarias. La regla es la siguiente: Si en un bucle hacemos uso de un dato relativo a un sensor y no hemos puesto una pieza para la observación del mismo, el programa la incluye automáticamente. No se ve en nuestro programa, pero está ahí.

Editando el fichero scribbler.ini podemos configurar estas observaciones automáticas según nuestros deseos, según la siguiente tabla:

autoreadsensors = 0	No inserta nada automáticamente
autoreadsensors = 1	Si no existe una observación y es necesaria la inserta. Si existe, no la inserta.
autoreadsensors = 2	Inserta siempre una observación automática de los sensores en el programa

2.8 Pieza de cálculo



La pieza de cálculo mostrada en la figura 33 se utiliza para realizar cálculos con las lecturas de los sensores de luz. Pero con el tiempo, en futuras versiones del software, se utilizarán para más cosas. La lectura actual del sensor de luz se representa en azul y las de referencia en rojo.

Existen 6 operaciones posibles. En la primera columna, a la izquierda, hay opciones para asignar los valores de referencia. La primera, la de arriba, asigna el valor 0 como valor de referencia. La segunda lee valores de referencia procedentes de una ROM y la tercera copia las lectura actual de los sensores como valor de referencia.

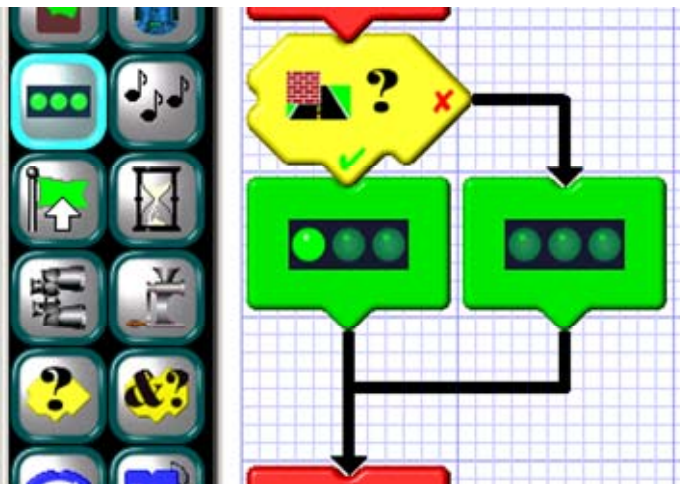
Figura 33

En la segunda columna se realizan diferentes cálculos entre los valores de referencia y las lecturas actuales de los sensores. La primera, la de arriba, resta el valor de la lectura actual con el de referencia. Si el resultado es menor que cero, éste se redondea y pone a 0. La segunda resta el valor de referencia con el valor de la lectura actual. Complementa la lectura actual y le suma el valor de referencia. Si el resultado es menor que 0 se redondea a 0. La tercera opción calcula el valor absoluto de la diferencia entre el valor en curso y el valor de referencia. El resultado es un valor positivo que se puede usar para detectar cambios en la intensidad de luz que incide sobre los sensores de un instante a otro.

3.- Piezas Condicionales

Las piezas condicionales se emplean para hacer un desvío en el curso de la ejecución del programa. Dependiendo de una condición el programa continuará por diferentes caminos.

3.1 Si-entonces



La condición más simple es la mostrada en la figura 34. (SI – ENTONCES- EN CASO CONTRARIO). Si la condición se cumple, el curso del programa será el marcado por el tic verde, y en caso de no cumplirse, el camino que seguirá el programa es el indicado por el aspa roja. En esta figura de ejemplo, **si** detecta un obstáculo **entonces** encenderá el LED de la izquierda y **en caso contrario** los LEDs quedarán apagados.

Figura 34

Las condiciones se insertan en el programa como cualquier otra pieza. Simplemente pinchamos en el botón condicional y aparecerá un cursor cuando movemos el ratón por el área de trabajo, presentándonos dónde se puede insertar la condición. Cuando hacemos clic en el sitio deseado, aparece una ventana para editar dicha condición, tal y como se muestra en la figura 35. Aquí podremos seleccionar qué condición queremos testear haciendo clic en uno de los 5 botones de opciones múltiples que aparecen. El último botón , abajo a la derecha, se usa para establecer las direcciones de Verdadero o Falso. Mediante la barra de scroll que aparece en la parte inferior de la pantalla de edición se puede establecer las magnitudes numéricas necesarias en alguna de las condiciones.

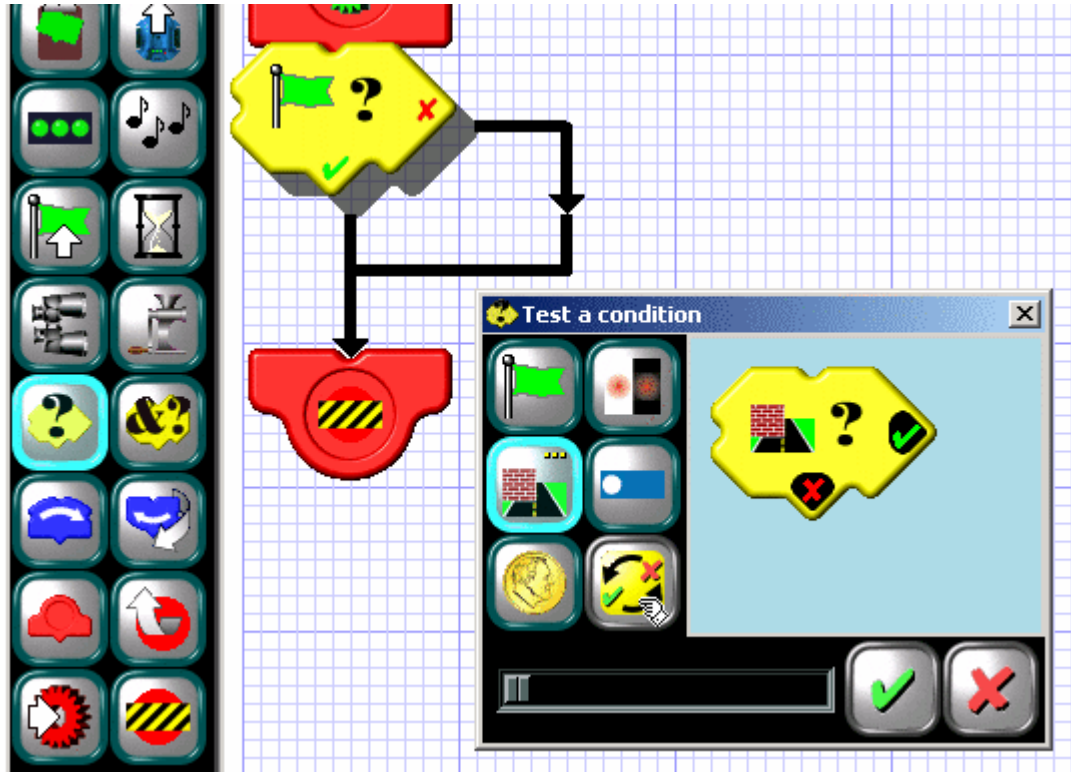
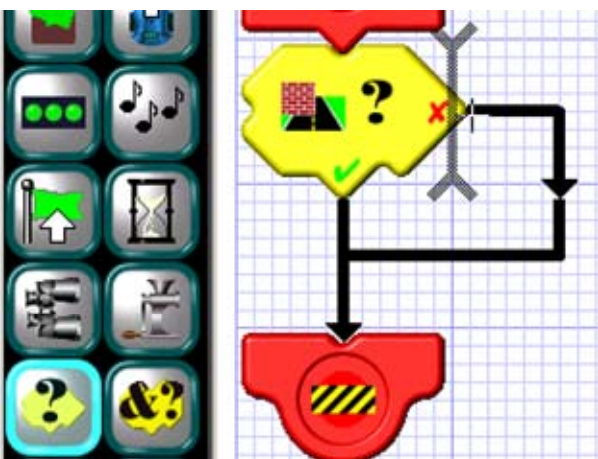


Figura 35

Cuando creamos una condición, se crean más sitios o puntos de inserción donde podremos colocar más piezas. Una es inmediatamente debajo de la pieza y normalmente se corresponde con el camino verdadero de la condición. En el ejemplo de la figura 35, hemos cambiado Verdadero y Falso, por lo que aquí es el camino FALSO el que está justo debajo de la pieza condicional. Otro punto de inserción de piezas es la flecha a la derecha que parte de la pieza de condición. Normalmente es el camino Falso, pero en el ejemplo, una vez más, es el contrario. Un tercer punto de inserción es debajo de la condicional. Este punto se utiliza cuando las direcciones distintas se vuelven a juntar. Es independiente de si el camino es Falso o verdadero.

3.1.1 En Caso Contrario



También existe otro punto de inserción posible, pero solo si vamos a incluir otra condición. Es justo a la derecha de la ficha condicional como muestra el cursor vertical de la figura 36.

Figura 36

Al insertar condiciones adicionales, se pueden crear múltiples caminos o rutas en la ejecución del programa, como en el ejemplo de la figura 37. Aquí, se testean todas las posibilidades de encontrarse con un objeto. Detectarlo a la izquierda, detectarlo a la derecha, detectarlo al frente no detectarlo. Cada posible condición tiene una acción distinta: Si se detecta obstáculo a la izquierda, se gira a la dcha.; si se detecta a la dcha. se gira a la izda.; si hay obstáculo se rota y, finalmente si no hay nada se avanza.

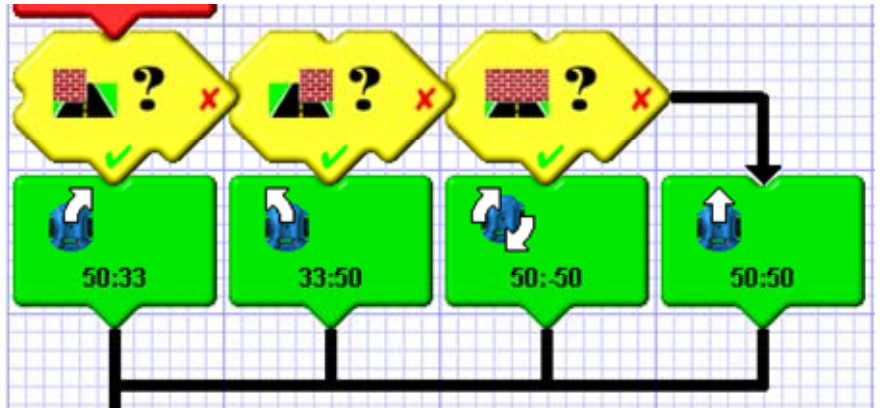


Figura 37

3.2 Y-Si



Si se necesita ser más restrictivo y añadir más condiciones adicionales, use la pieza Y-Si. Esta solo puede ser insertada debajo de una pieza de condición o debajo de otra pieza Y-si. En el ejemplo de la figura 38, el LED de la izquierda se encenderá si el sensor de línea no detecta una línea negra y si el aviso o bandera verde no está levantada y si el tiro de la moneda resulta “cara”. Esas 3 condiciones deben cumplirse para que la condición sea cierta y tan solo en ese caso se encenderá el LED. El tema del tiro de la monedas puede resultar un tanto extraño. Realmente no existe moneda alguna, es solo un símil que genera una condición aleatoria con dos posibles estados: “cara” o “cruz”.

Figura 38

3.3 Condición de Aviso por Bandera

Existen 7 diferentes banderas de diferentes colores y con 2 posibles estados, levantada o bajada. La condición puede chequear banderas para ver si están levantadas o no. En el ejemplo de la figura 39, el LED izquierdo refleja la situación de la bandera roja. Si está levantada se encenderá y si está apagada se apagará. Las banderas se pueden usar a modo de variables.

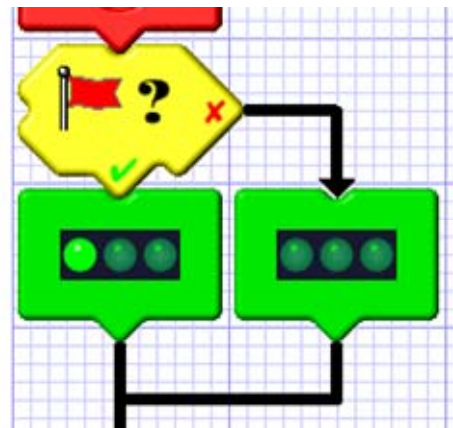


Figura 39

3.4 Condición del sensor de líneas

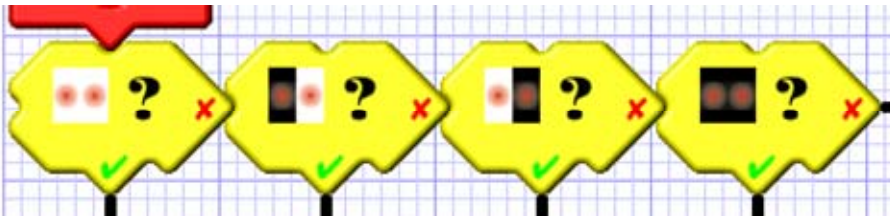


Figura 40

El sensor de líneas utiliza dos detectores ópticos en la parte inferior del robot capaces de detectar marcas negras y blancas en el suelo. Se utiliza principalmente para el seguimiento de líneas o lectura de códigos de barras.

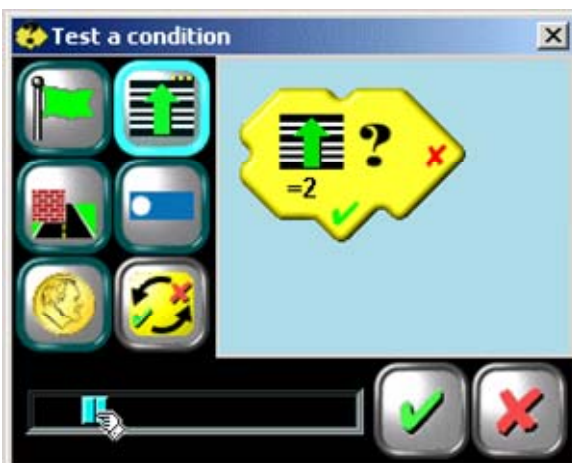
Tienen 4 posibles estados: sin línea negra visible, línea negra solo en la izquierda, línea negra sólo en la derecha y línea negra en ambos. El ejemplo de la figura 40 muestra los 4 posibles estados.

También podemos hacer más restrictiva la condición del sensor, requiriendo un cierto número (hasta 8) de observaciones consecutivas TRUE. Esto se hace con la barra de scroll de la parte inferior de la pantalla de edición de condiciones. En el ejemplo de la figura 41 lo hemos desplazado hasta exigir 3 ocasiones en las que se detecte la línea negra por la izquierda antes de que se cumpla la condición global.



Figura 41

3.5 Sensor de códigos de barra

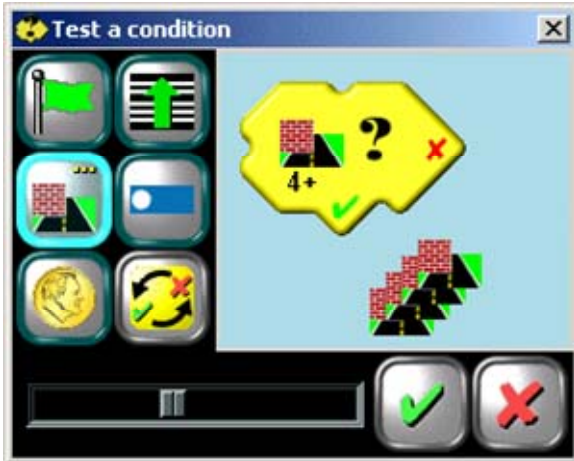


El sensor de línea también se puede usar para leer códigos de barra. Se pueden identificar hasta 7 códigos de barra que están disponibles en la documentación del robot. Necesitará el Acrobat Reader para imprimirlos. Es decir, no reconoce cualquier código de barras, sino tan solo los 7 que acompañan a la documentación en un pdf.

Debemos ser cautos y no hacer que el robot vaya muy rápido cuando queramos que lea códigos de barra. Se pueden usar códigos de barra para hacer laberintos de forma que cada código de barra indique al robot por donde debe ir. Eso sí, no podemos usar la función de seguimiento de líneas y de códigos de barra simultáneamente ya que utilizan el mismo sensor. En la figura 42 la condición se establece cuando el robot, en su avance, reconozca el código de barras correspondiente al nº 2

Figura 42

3.6 Detector de obstáculos



El sensor de obstáculos consta de 2 sensores infrarrojos con los que es posible detectar objetos en la parte delantera izquierda y derecha. Como el sensor de líneas, esta tiene 4 posibles estados: Objeto a la izquierda, a la derecha, al frente, o en ninguno. En el ejemplo de la figura 43 la condición será cierta cuando se detecta un objeto por la izquierda en cuatro ocasiones consecutivas

Figura 43

3.7 Detector de choque

El sensor de choque detecta cuándo las ruedas del robot se están moviendo pero el robot se desplaza. Con ello se deduce (aunque no siempre debería ser así) que ha colisionado con un objeto. Podemos restringir este sensor a que suceda varias veces (2 en la figura 44). Es conveniente usar este sensor junto con los sensores de detección de obstáculos para saber que no se mueve porque hay un objeto delante.

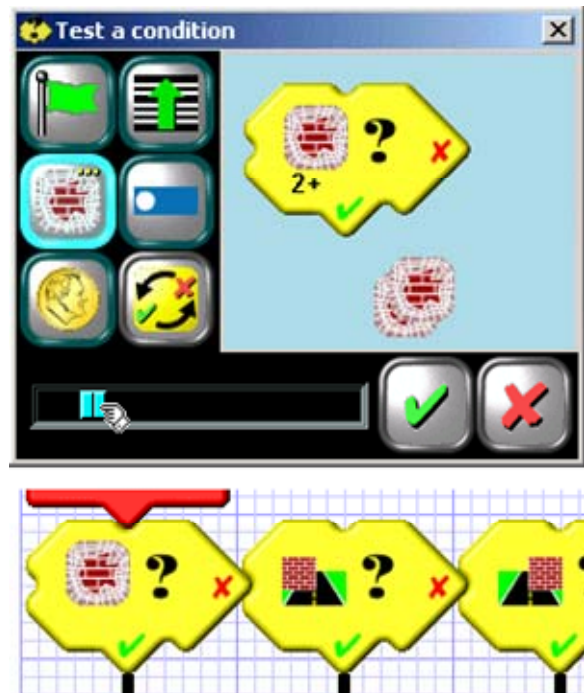
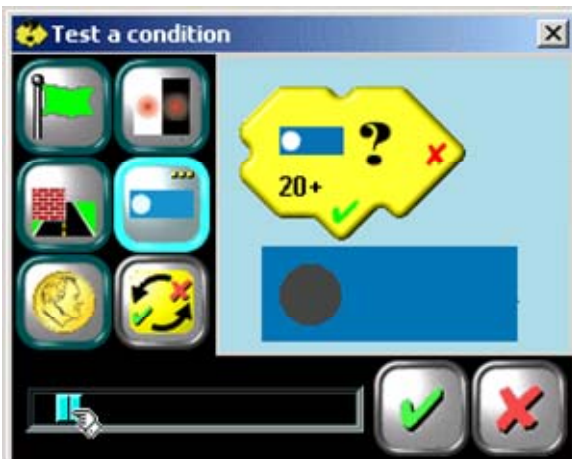


Figura 44

3.8 Condición del sensor de luz



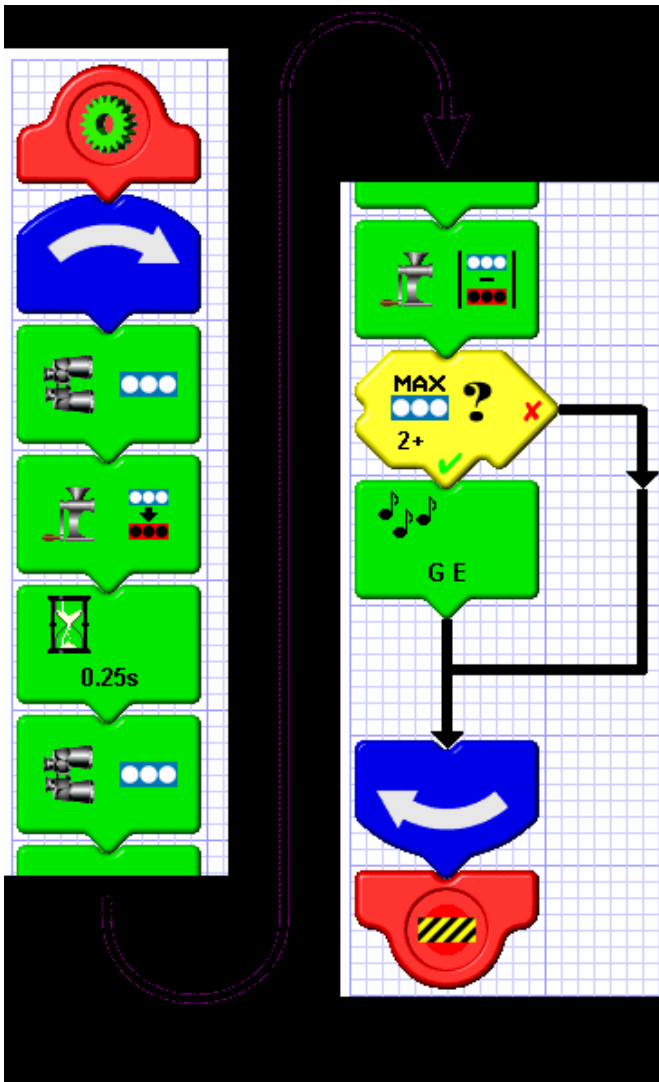
El sensor de luz consiste en 3 fotocélulas montadas bajo el chasis del robot que detectan la luz por unos agujeros en la parte delantera del mismo. La cantidad de luz detectada se compara con distintos valores almacenados. Se pueden validar 12 condiciones agrupadas en 4 categorías:

- 1.- El valor del sensor (izq.,der. o central) es menor que X.
- 2.- El sensor izq.,der. o centro es el más brillante de los 3 en al menos X.
- 3.- El sensor izq.,der. o centro es el más oscuro de los 3 en al menos X.
- 4.- El máximo, mínimo o media del sensor es al menos X.

Figura 45

El valor de X puede ir de 1 a 255 y se ajusta con la barra de scroll. En el ejemplo de la figura 45 el sensor izquierdo debe tener un valor de al menos 20 para que la condición sea cierta.

Analicemos como funciona el programa de la figura 46 que se muestra a continuación



1. Se lee el valor actual del sensor de luz.
2. Los valores leídos se guardan como referencia
3. El programa hace una pausa de ¼ de segundo.
4. Se vuelve a leer el sensor de luz.
5. Se calcula el valor absoluto entre la diferencia de la última lectura y la de referencia.
6. Si el valor más alto de alguno de los 3 es al menos 2 se produce un pitido

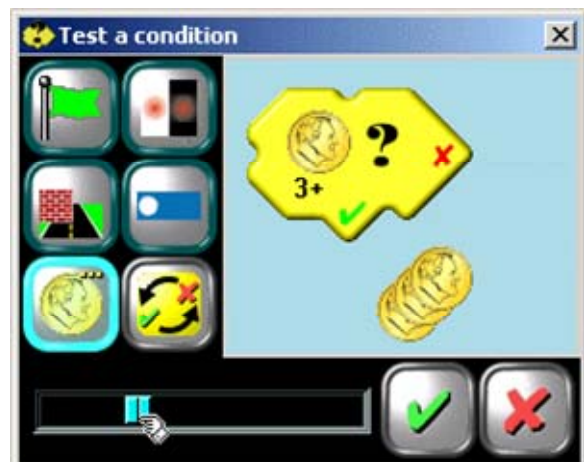
Al cargar este programa en el robot, lograremos que el robot pite cada vez que alguien pasara por encima de él y le quitara algo de luz.

Figura 46

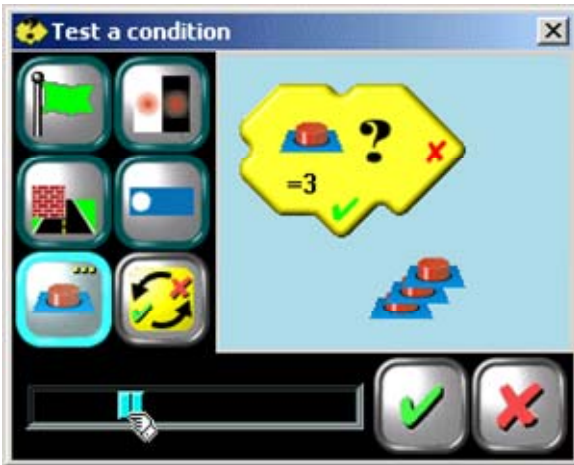
3.9 Lanzamiento de monedas

Esta condición permite añadir cierta característica de azar tipo "cara o cruz" en nuestros programas. En el ejemplo de la figura 47, la condición se activará cuando salgan 3 caras seguidas. El número de caras que queramos conseguir es inversamente proporcional a la probabilidad de que salga VERDADERO.

Figura 47



3.10 Pulsación del botón de reset



Esta condición chequea si se ha pulsado el botón de reset (o si se ha encendido el botón de ON/OFF). Al presionar el botón de reset se reinicia el programa. Luego esta condición nunca puede ser cierta excepto al inicio de programa y se suele utilizar para hacer distintas cosas dependiendo del número de veces que se haya pulsado el botón de reset. Así en el ejemplo de la figura 48, se establece la condición de si se ha pulsado el botón de reset 3 veces. Naturalmente, insistimos en que esta condición sólo tiene sentido si es la primera de nuestro programa.

Figura 48

En el ejemplo de la figura 49, el programa hace lo siguiente:

Si se pulsa el botón de reset 3 veces, el robot avanza durante 3 segundos y en caso contrario, no hace nada. Así el robot acaba su ejecución y hasta que no volvamos a pulsar 3 veces seguidas el reset, se quedará parado.

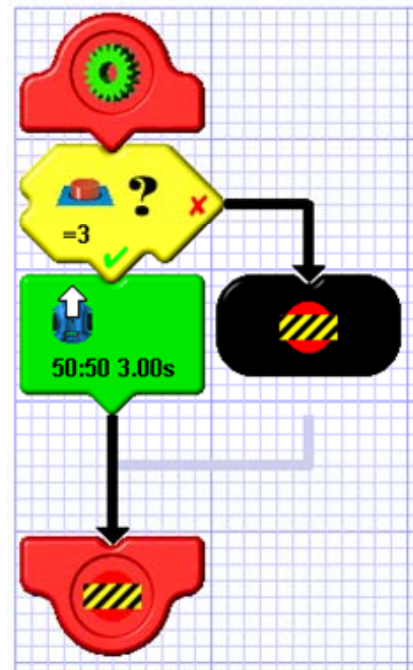


Figura 49

4.- Bucles

Son mecanismos por los cuales es posible repetir la ejecución de varias instrucciones o piezas dentro de un programa.

4.1 Repeticiones tipo loop



Las piezas de repetición Loop, nos permiten repetir la ejecución de unas piezas de programa. Siempre se insertan por pares (figura 50), para delimitar como si fueran 2 paréntesis, la zona que queremos repetir. Primero insertamos el loop y después insertamos dentro las acciones que queramos repetir. Pueden ser finitos o infinitos. Los infinitos se repiten siempre y los finitos se repiten un número determinado de veces (hasta 255). En el ejemplo de la izquierda, el bucle se repite infinitas veces.

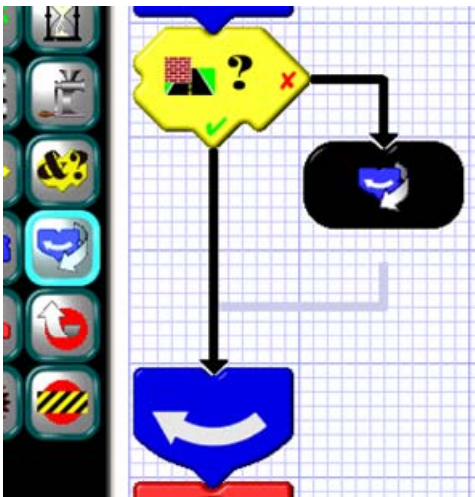
Figura 50



Figura 51

Cuando insertamos un bucle nos aparece la pantalla de configuración del bucle. Al principio saldrá oscura, si la dejamos así el bucle será infinito. Si acercamos el ratón a los números, observaremos cómo podemos cambiar el número de bucles y así determinar el número de repeticiones. Ver la figura 51.

4.2 Etiquetas de salida



Cuando estamos dentro de un bucle infinito o finito, quizás sea interesante tener la posibilidad de salir de dicho bucle. Para ello existen las etiquetas de salida. Estas se colocan en una condición dentro de un bucle como el de la imagen de la figura 52 Cuando no se cumpla dicha condición, se terminará el bucle, pero mientras se cumpla el bucle se estará ejecutando indefinidamente.

Figura 52

5.- Subrutinas

Las subrutinas son segmentos de acciones semi independientes que pueden ser llamados desde otras partes del programa. Cuando una subrutina es llamada, el programa que llama a dicha subrutina, suspende temporalmente la operación para que esta se ejecute. Cuando esta acaba su trabajo devuelve el control al programa que la llamó.

5.1 Creando Subrutinas

En el generador de programas scribbler, puede disponer de 8 subrutinas diferentes distinguidas mediante un engranaje de distintos colores, incluido el programa principal que es verde. Las subrutinas tienen siempre dos piezas, una de comienzo y otra de fin. Las de color verde corresponden al programa principal y las demás son amarillas, naranjas, rosas, magenta, púrpura, azul y cian, en orden de importancia. Ver la figura 53



Figura 53



Hay dos formas de añadir subrutinas a un programa. La primera simplemente pinchando en el botón de añadir subrutina mostrado en la figura 54. Y la segunda es simplemente llamando a dicha subrutina. Puede haber un máximo de anidación de 3 niveles.

Figura 54

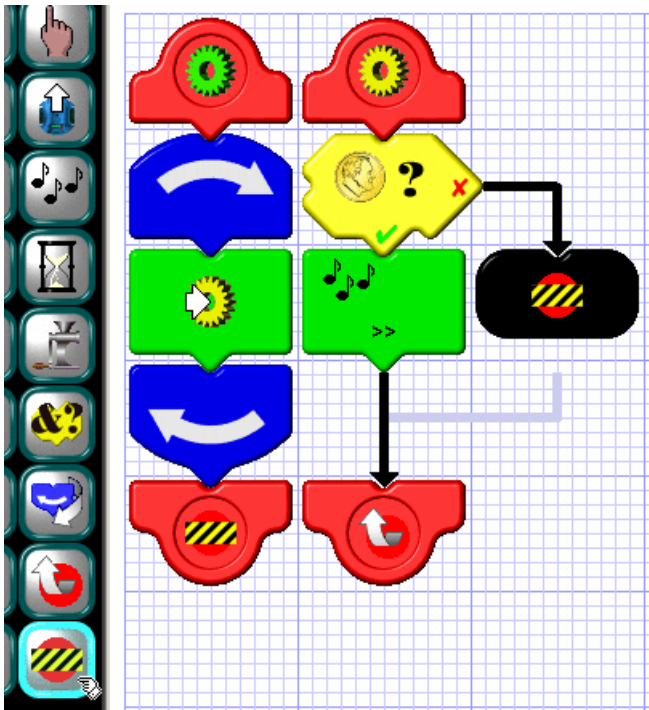
Existen etiquetas de retorno para volver de la subrutina al punto desde la cual se le llamó. Y etiquetas de fin de programa para acabar la ejecución de un programa.

Como se muestra en el ejemplo de la figura 55, el programa principal (representado por el engranaje verde) consiste en un bucle infinito que llama constantemente a la subrutina amarilla.

Esta subrutina a su vez comprueba si hay un obstáculo a la izquierda del robot en cuyo caso lo hace rotar a la derecha. La comprobación se realiza de forma constante en otro bucle infinito que sólo finaliza cuando no se detecte el obstáculo. En ese momento el robot avanza, finaliza el bucle y finaliza la subrutina.

Se retorna al programa principal y el proceso se vuelve a repetir.

Figura 55



En el ejemplo mostrado en la figura 56 el programa principal llama a la subrutina amarilla.

La subrutina se encarga de "tirar la moneda". Si sale cara se genera una melodía y si no finaliza su ejecución retornando al programa principal.

El proceso se repite de forma indefinida.

Figura 56

6- Sencillo ejemplo de rastreo

El ejemplo mostrado en la figura 57 trata de hacer que el Scribble siga o rastree una línea negra pintada en el suelo.

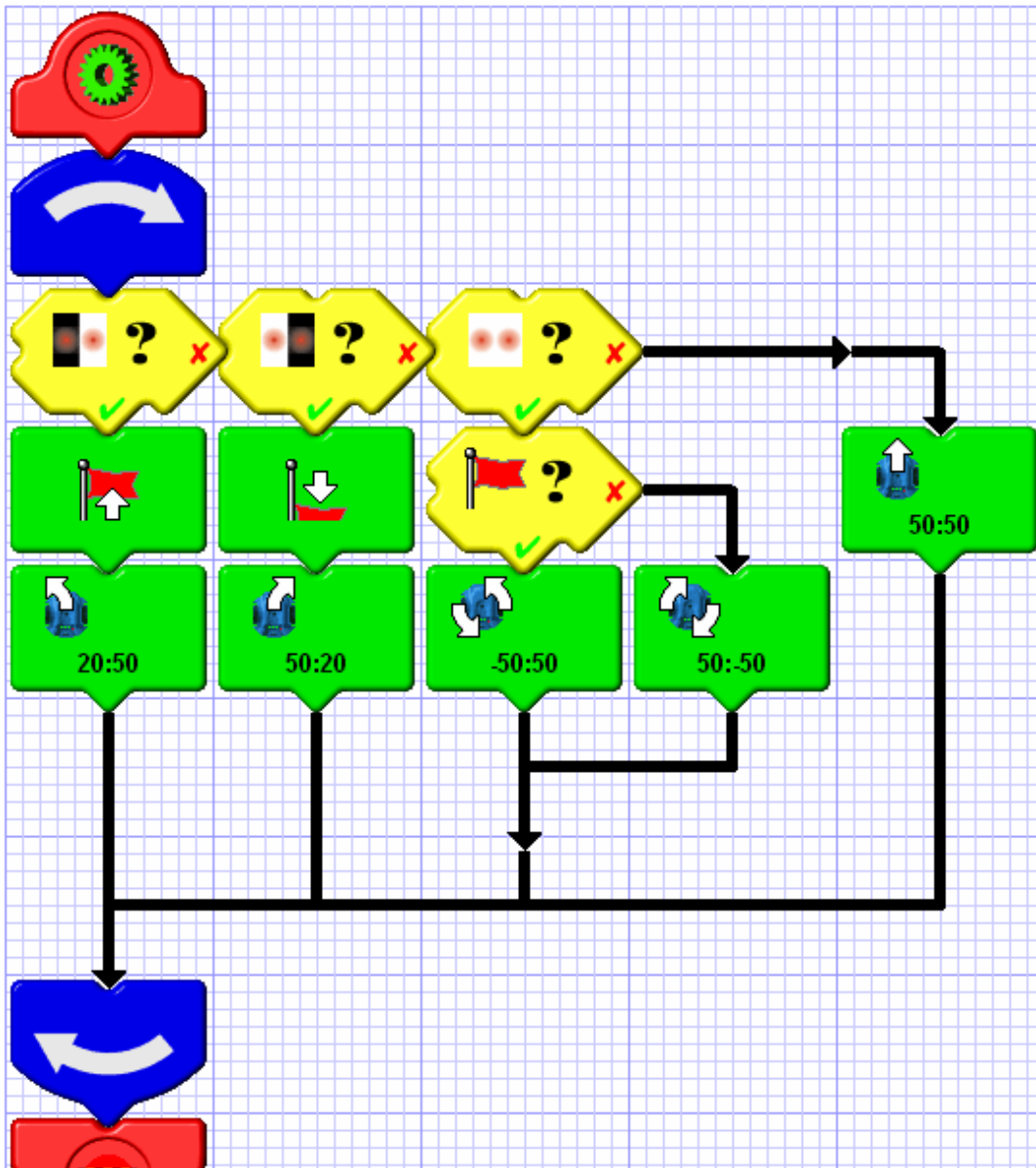


Figura 57

El programa comienza comprobando si el robot se ha salido por la derecha de la línea negra, en cuyo caso se levanta la bandera roja y se gira a la izquierda.

En caso contrario se comprueba si se ha salido por la izquierda. De ser así se baja la bandera roja y se corrige girando a la derecha.

También se comprueba si el robot está totalmente fuera de la línea. De ser así y, si la bandera roja está levantada, se realiza una rotación a la izquierda y si la bandera está bajada la rotación es hacia la derecha.

Finalmente, si los sensores del robot no detectan ningún tipo de desvío, es decir están sobre la línea negra, se realiza un movimiento de avance.